LINNET

A SMALL-SIGNAL A. C. NETWORK ANALYSER

FOR THE 48K SINCLAIR SPECTRUM

COPYRIGHT CHEZ WATTS

1983

INTRODUCTION

LINNET can be used to analyse circuits operating in the linear region that are composed of networks of resistors, capacitors, inductors, transformers, transistors and amplifiers. All components are considered ideal (or nearly so), but non-ideal components can be modelled by combining several ideal elements. The program features both time domain and frequency domain analysis and subcircuit calls and is menu driven.

Internally the program does not create a large matrix of simultaneous equations but instead converts each element into its equivalent G matrix and combines this with the G matrix that represents the circuit so far. This method, although it does have some limitations (see below), leads to a compact notation for entering the network description and allows much larger networks to be tackled in a given memory space.

The method and notation were described by Colin Gyles in Electronic Design 9, April 25, 1980 pp191 - 197 and the reader is referred to this text for a full description of them.

LIMITATIONS

LINNET can only analyse linear circuitry, therefore it cannot be used to analyse Class B,C or D amplifiers which all utilise non-linear elements, neither can it be used to analyse switching or digital circuitry or DC bias conditions. In addition, although the network description notation allows most circuits to be described it is not completely universal so that it is not possible to analyse all network topologies. In practice this is not likely to be a severe limitation since most practical networks can be changed to a near equivalent which can be analysed.

A valid circuit may occasionally fail in analysis with a division by zero error. In this case it is usually possible to find an alternative ordering of the circuit elements which avoids this error.
Large networks comprising several hundred elements can be analysed on the 48K Spectrum. In practice the User is likely to find the time the computation takes the major limitation and is unlikely to be constrained by memory limitations.
It is not possible to protect against every possible input error, without excessive use of memory, with a program as complex as LINNET. If at any time you find that you have caused the computer to stop with an error report, entering "GOTO 9250" will re-establish control.

METHOD

A circuit diagram is drawn of the circuit to be analysed in conventional format (i.e. input to the left, output to the right) An AC equivalent of this is then drawn in the same format. That is the power rail is treated as ground and any capacitors whose reactance is negligible at the frequencies of interest are replaced by short circuits and any inductors whose susceptance is negligible at the frequencies of interest are replaced by open circuits. The AC equivalent circuit is then divided into elements of the kind to be described and the juxtapositions of the elements noted so that they can be entered in the programs notation. With a little practice the optimum order of the element in the element list can be quickly spotted and the element list can be typed into the computer directly by inspection of the circuit diagram. Occasionally the circuit will need to be redrawn for the element list to become clear and sometimes pseudo-elements (such as zero resistance resistors) or the creation of subcircuits may be required so that the topology yields to the method. Study the examples given to see how these tricks are implemented in practice.

Three types of input data are required:-

1. Circuit description. The components, their values, and how they are connected.

2. The range of frequencies over which the analysis is to be performed.

3. A description of the output required and the format it is required in.

The program is menu driven and the User is prompted for each piece of data in turn. After all the data is entered the User is given the opportunity to LPRINT the input data to check its correctness and change any errors. The analysis can then be executed and the results are output. The output data is stored internally in an intermediate form so that the output requests can then be altered and the output routines re-run. This allows, for example, the plots to be output to the screen and if they appear satisfactory to be plotted at greater resolution later onto the ZX printer without again executing the analysis.

ELEMENT DESCRIPTION

Elements are described by two letter mnemonics, one to identify the type of element and one to describe its coupling configuration with the rest of the circuit plus a real number (two in the case of a transistor) for the elements value. The mnemonics must be in the right order and separated by single spaces. The value may be input in any form acceptable to ZX BASIC as a real number but must not be more than 18 characters - this includes both numbers and their space separator in the case of the transistor.

ELEMENT SYNTAX

Most elements:-

```
         ( rs )
         ( rb )                ( st )
         ( cs )                ( lt )
         ( cb )                ( rt )              value
         ( ls ) 1 space        ( ab ) 1 space   (18 chars
         ( lb )                ( be )              max)
         ( tf )                ( ip )
         ( am )
         ( at )
```

Examples:-

```
rs st 4700
cb lt .000000000022
am ip -10
ls ab 6.8e-3
rb rt le6
tf be 33
```

Transistors:-

```
         (st)
         (lt)      Ic in mA                  Current gain
   tr    (rt)
         (ab)      (Both negative for common
         (be)      emitter, both positive for
         (ip)      common base)
```

Examples:-

```
tr lt 10 .97
tr rt -24 -650
```

SUBCIRCUITS

Subcircuits are described in the same syntax. They start with an element with the st mnemonic and end with the mnemonic en followed by one space and any suitable name (18 characters max - no spaces). Up to 20 subcircuits are allowed.

Examples:-

```
rs st 2200
cb rt 1e-9
rs rt 2200
en pisection
```

(HF transistor model with parasitics)

```
tr st 20 240
rb ip 6e6
cb ip 12e-12
cs lt 4e-12
rb lt 15
en hftransistor.
```

SUBCIRCUIT CALL

The subcircuit is called using the name given it in the en statement. The mnemonic for call is cl and to these two must be added the mnemonic that describes the coupling of the subcircuits.
Subcircuits may be called any number of times but the subcircuit description must occur before any calls to it in the element list.

SUBCIRCUIT CALL SYNTAX

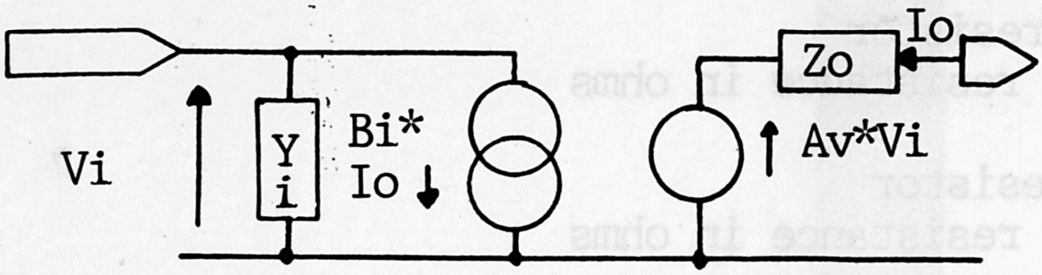
```
                                 ( st )
                                 ( lt )
                                 ( rt )
    cl  (namewithoutspaces)      ( ab )
                                 ( be )
                                 ( ip )
```

```
Examples:-          cl hftransistor ip
                    cl pisection rt
```

PERMISSABLE ELEMENTS

All elements are converted internally into three terminal two port networks characterized by G parameters. The general equivalent circuit is shown in fig. 1. This leads to two forms of circuit element each for the passive elements resistors, capacitors, and inductors called here the "shunt" and "bridge" forms. The circuits of all the permissible elements are shown in fig. 2.

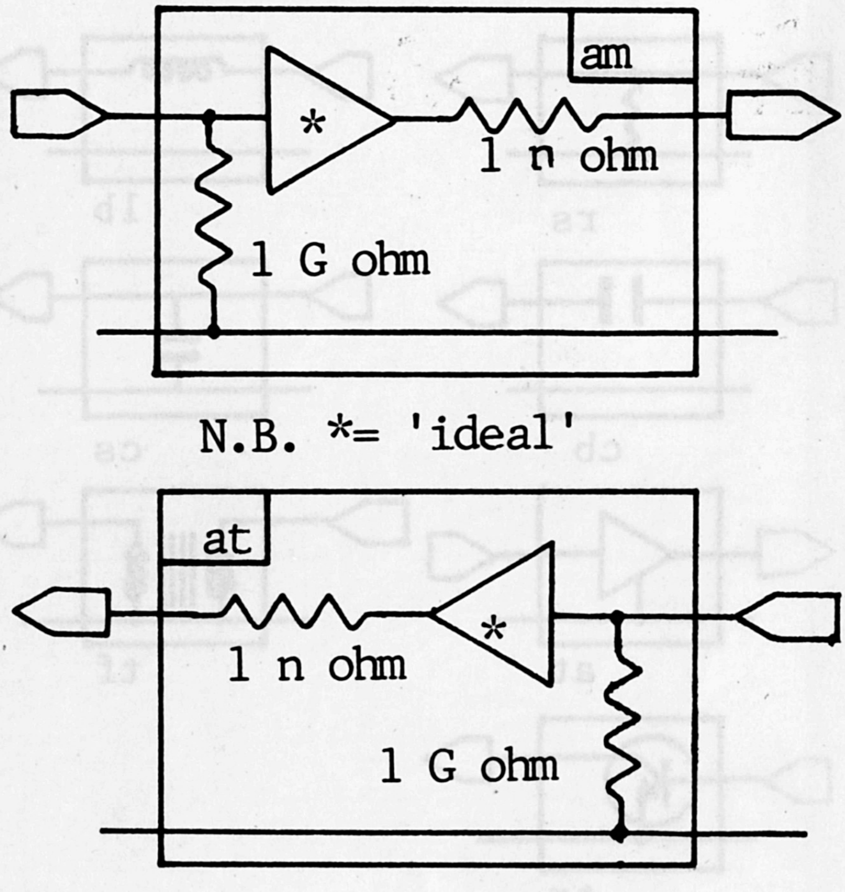
FIG. 1. THE "G" PARAMETER CIRCUIT.



All elements are conveted into this general form by the program which then combines them to form a single "G" parameter equivalent at any frequency.

Av = forward voltage gain.
Bi = reverse current gain.
Zo = output impedance.
Yi = input admittance.

FIG. 3. AMPLIFIER MODELS



These are given finite I/O impedances as shown here. This reduces the risks of program failure caused by division by zero errors occuring.

n.b.:-
1 G ohm = 10
9 ohm

1 N ohm = 10
-9 ohm

There are also two forms of the amplifier element, one with its input and output transposed (at) in order that amplified or isolated feedback may be achieved.

rb bridge resistor
value = resistance in ohms

rs shunt resistor
value = resistance in ohms

cb bridge capacitor
value = capacitance in farads

cs shunt capacitor
value = capacitance in farads

lb bridge inductor
value = inductance in henries

FIG. 2. ELEMENT TYPES.
All the two port three terminal network elements that are shown here may be used in LINNET.

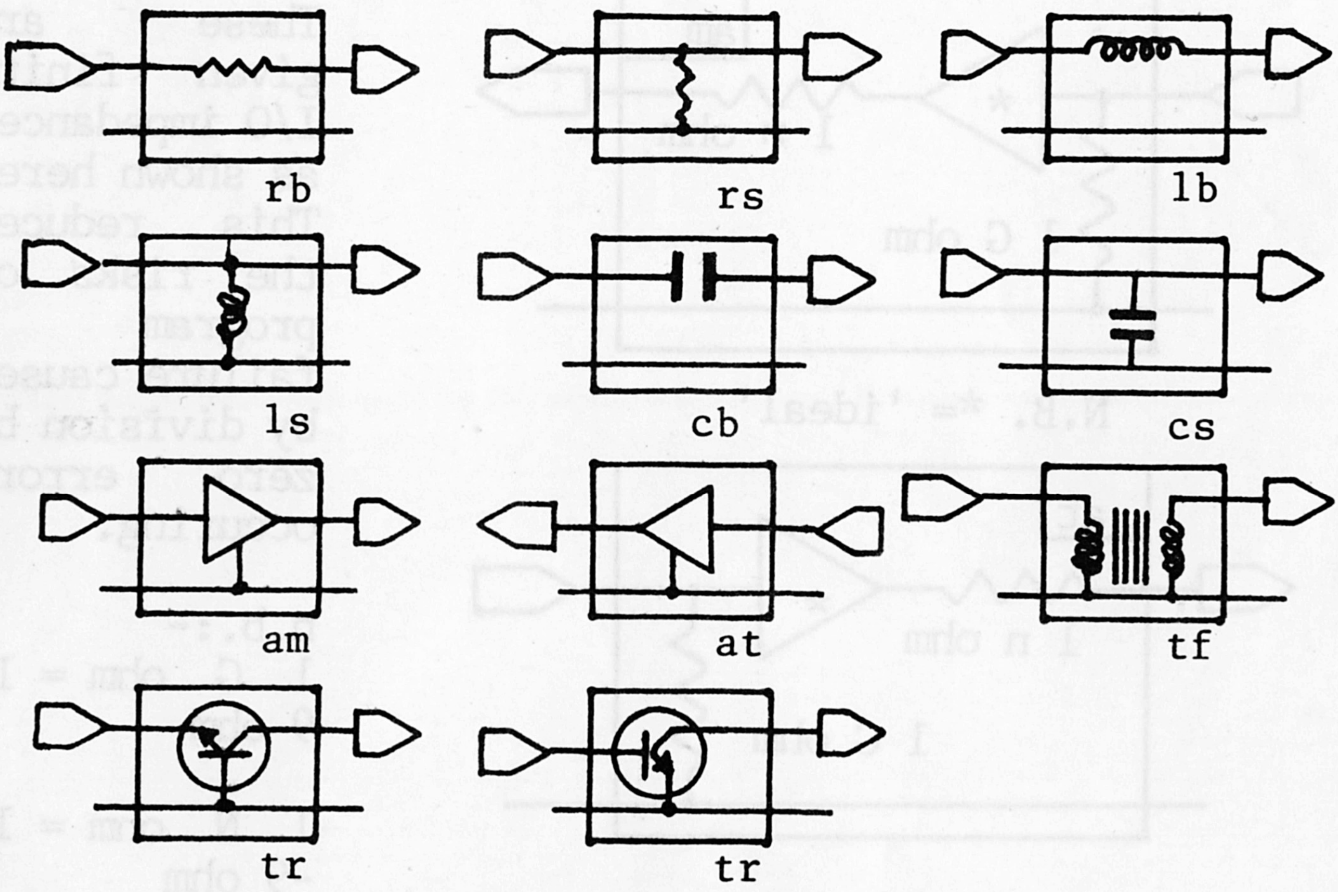
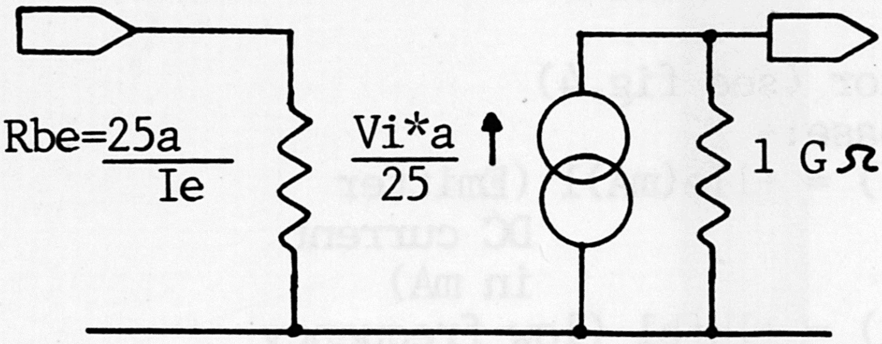


FIG. 4. LINNET'S TRANSISTOR MODEL



*LOW FREQ MODEL	Common base	Common emitter
a = current gain	hfb +ve	hfe -ve
Ie = DC emitter current	+ve	-ve

Note: gm = .04*Ie
Rbe = a/gm
hfb = 1/(1+hfe)
*(For high frequencies add external parasitics)

ls shunt inductor
value = inductance in henries

am amplifier (see fig 3)
input impedance = 1G ohm
output impedance = 1n ohm
value = voltage gain.

at amplifier (input and output transposed - see Fig 3).
input impedance = 1G ohm
output impedance = 1n ohm
value = voltage gain

tf   transformer
      value = turns ratio

tr   transistor (see fig.4)
      common base:-
      value (1) = +lIe(mA)l (Emitter
                             DC current
                             in mA)
      value (2) = +lhfbl (low frequency
                          current gain
                          =l/(l+hfe))
      common emitter:-
      value (1) = -lIe(mA)l (Emitter DC
                             current in mA)
      value (2) = -lhfel (low frquency
                          current gain)

COUPLING CONFIGURATIONS

The five configurations provided for coupling a new element into the circuit so far are shown in Fig 5. The sixth mnemonic (st - start) is used when the element is the first in a circuit or sub-circuit. Figs. 6 - 10 show how a sample network may be partitioned into three terminal networks and coded up and how the program merges the elements together in the coding order to finally produce a G parameter equivalent to all the elements together at one frequency.
The coupling configurations are as follows:-

st   start - used for first element in
              circuit or subcircuit.

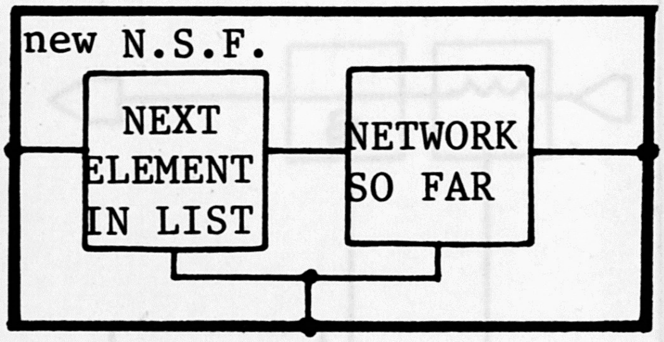
lt   to left - the new element is added to the left of the previous elements.

rt   to right - the new element is
                 added to the right of
                 the previous elements.

ab above - the new element is added above the previous elements.

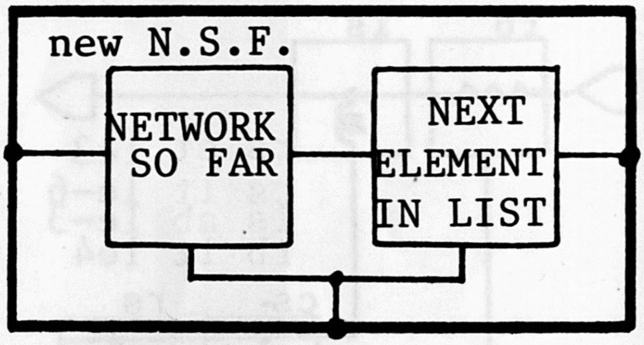
be below - the new element is added below the previous elements.

ip in parallel - the new element is added in parallel with the previous elements.



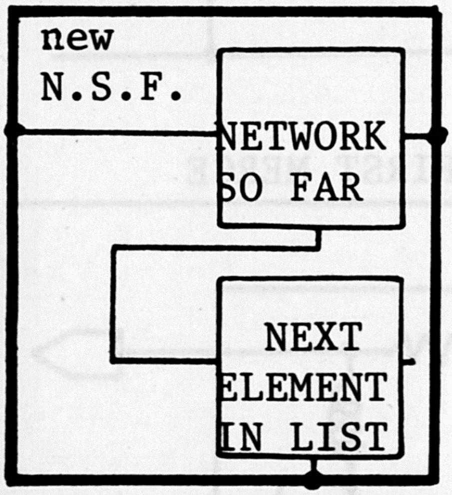
TO LEFT lt

FIG.5A



TO RIGHT rt

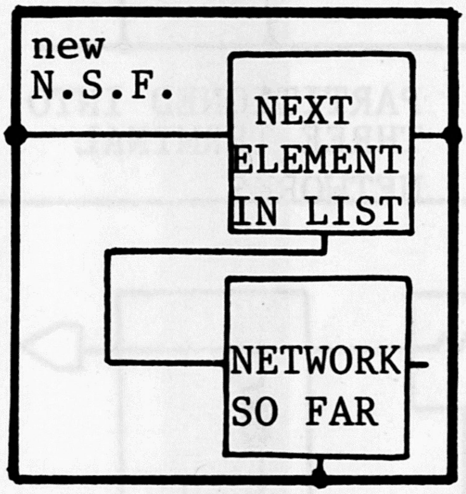
FIG.5B



BELOW

be

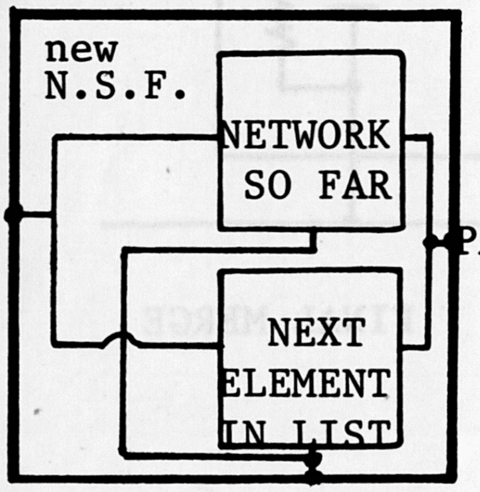
FIG.5C



ABOVE

ab

FIG.5D



IN PARALLEL

ip

FIG.5E

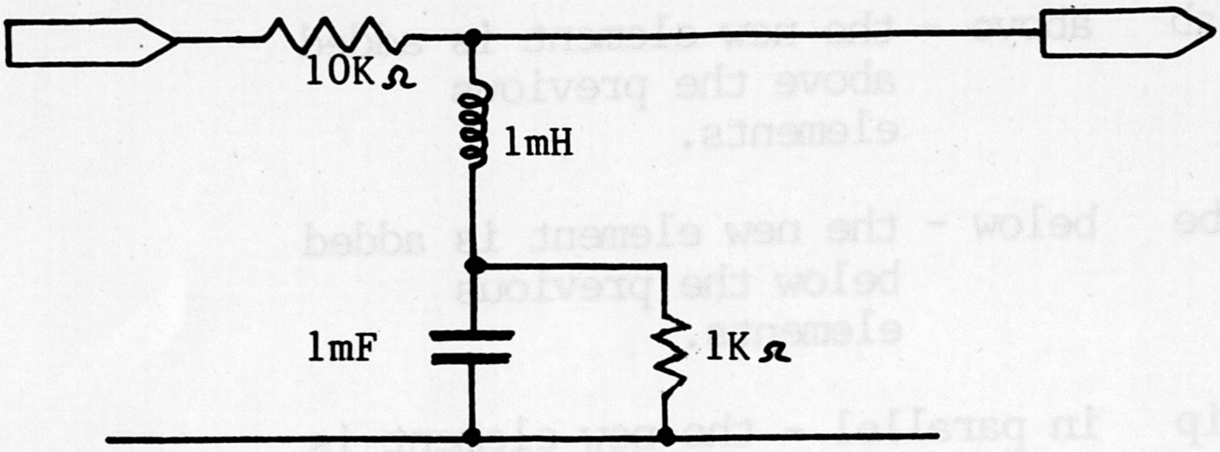


FIG.6 A NETWORK CODING EXAMPLE

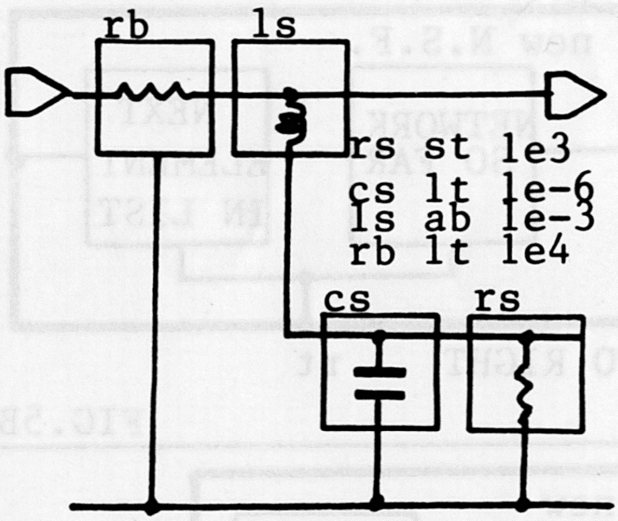


FIG.7 PARTITIONED INTO THREE TERMINAL NETWORKS

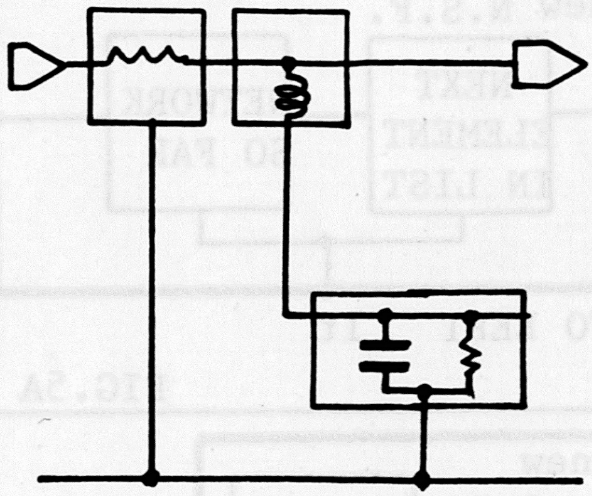


FIG.8 FIRST MERGE

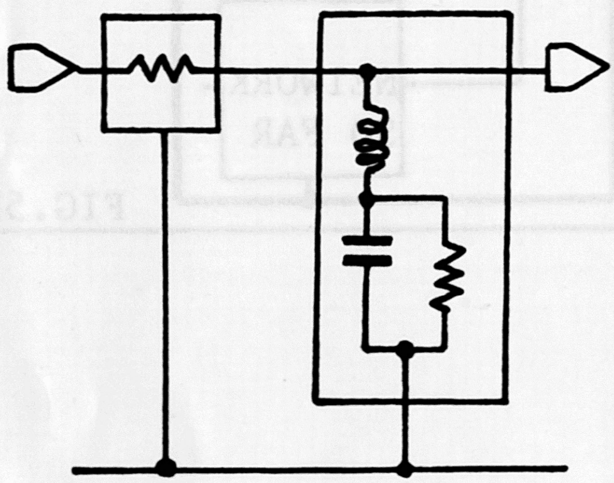


FIG.9 SECOND MERGE

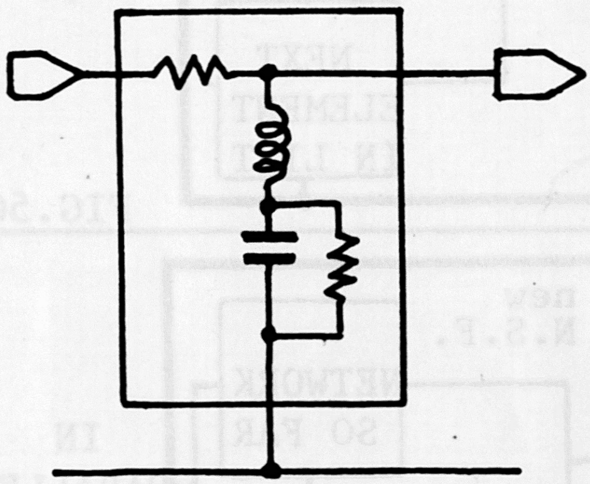


FIG.10 FINAL MERGE

OUTPUT REQUESTS

The output requests inform LINNET which data you require from the analysis and the format that you require it in. They take the form of a two letter mnemonic for the kind of information (voltage gain, output impedance etc.), a two letter mnemonic for the output presentation (plot, tabulate etc.) and a three letter mnemonic for the output format (real/imaginary, magnitude/phase etc.).

Syntax:-

```
( Av )
( Bc )
( ZO )                 ( pw )
( Yi )   1 space       ( tb )   1 space   ( rec )
( Ac )                 ( pl )             ( lna )
( Bv )                 ( ps )             ( lga )
( Zi )
( Yo )
```

Examples:- Av pl lga
Zi ps rec
Yo tb lna

(N.B. The first mnemonic of the output request MUST start with a capital letter).

There is no limit to the number of output requests in any analysis. Since the outputs are stored in intermediate form the output request list can be altered or added to and the output request processing part of the program re-run without re-executing the analysis providing the frequency range and step remain unaltered.

Zo output impedance with input short circuited = Vout/Iout.

Zi input impedance = Vin/Iin

Yo output admittance with input short circuited = Iout/Vout

Yi input admittance = Iin/Vin

Av forward voltage gain = Vout/Vin

Ac forward curent gain = Iout/Iin

Bv backward voltage gain = Vin/Vout

Bc backward current gain = Iin/Iout

OUTPUT PRESENTATION
----

tb tabular output
Data is output to printer in three columns (freq. and the two components of the complex parameter as requested in the format mnemonic). All numbers are in exponential format with six figures of mantissa and two of exponent.

pl plot (long)
Data is output to printer in the form of a long plot (frequency axis along the length of the paper). The plot is scaled on both axes and has vertical and horizontal divisions drawn.

pw plot (wide)
Data is output to printer in the form of a wide plot (frequency axis across the width of the paper). The plot is fitted to a rectangle with

unscaled axes although maximum and minimum values of both curves are printed alongside.

ps plot (to screen)
Data is output as for pw but to screen only with option to copy on the printer after the data has been seen.

OUTPUT FORMAT

rec rectangular format. The output data is presented in real/imaginary form.

lna linear amplitude. The output data is presented in magnitude/phase form.

lga log amplitude. The output data is presented in magnitude/phase form where the magnitude as presented =20*log(10) (true magnitude) thus enabling circuit gains to be ouput directly in decibels.

FREQUENCY RANGE OF ANALYSIS

After the element list and output requests have been entered LINNET prompts for information connected with the required frequencies at which the analysis is to be performed. First it enquires whether a time domain (see below) or frequency domain analysis is required. Where a frequency domain analysis is requested it then asks whether log. or lin. frequency increments are desired. In the case of linear increments it then prompts for the frequency increment per calculation step, the lowest and the highest frequencies. For logarithmic increments it prompts for the number of frequency increments per octave or per decade, the lowest and highest frequencies.

TIME DOMAIN ANALYSIS

Time domain analysis may occasionally be useful when designing wave-shaping circuits (e.g. at A to D interfaces) or circuits that have to propogate known waveforms with the minimum of phase dispersion. It may also be useful for rapidly checking LINNETS results against real circuits where frequency sweep apparatus is not available. To execute time domain analysis LINNET performs a Fourier transform on the waveform input by the user, executes a frequency analysis on the Fourier components and then does a second Fourier transform on these results to generate the output waveform.

If the time domain option is selected LINNET first asks for the order of Fourier transform. Low orders are inaccurate but high orders are very time consuming computationally. The User then specifies whether he wishes to enter the waveform in a piece-wise-linear format or expression form. After the waveform is input it is plotted and the user is asked to confirm that it looks correct. When it is correct the Fourier analysis is performed and the frequency spectrum of the input waveform is plotted. If an analysis is now executed LINNET will plot the frequency spectrum of the output waveform and the output waveform itself.

APPENDIX

EXAMPLES OF NETWORK CODING

Here are a few examples of practical networks and the corresponding element list suitable for LINNET input. In most cases the list is not the only feasible one for that circuit. Where there is an option the lt and be coupling configurations are preferred to the rt and ab configurations since they run slightly faster.

ACTIVE BANDPASS FILTER

See Fig.11.

```
am st 1e5
rb ip 1e5
cb lt 68e-9
cb ip 68e-9
rs lt 1540
rb lt 12400
```

FIG. 11 Active Bandpass Filter
(Centre frequency 200 hertz)

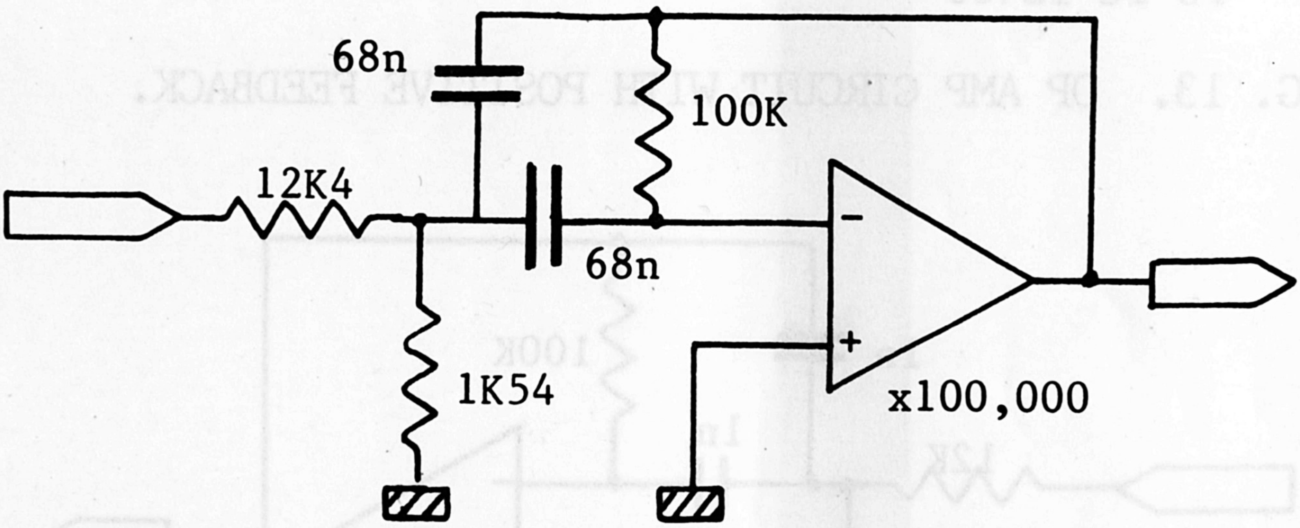
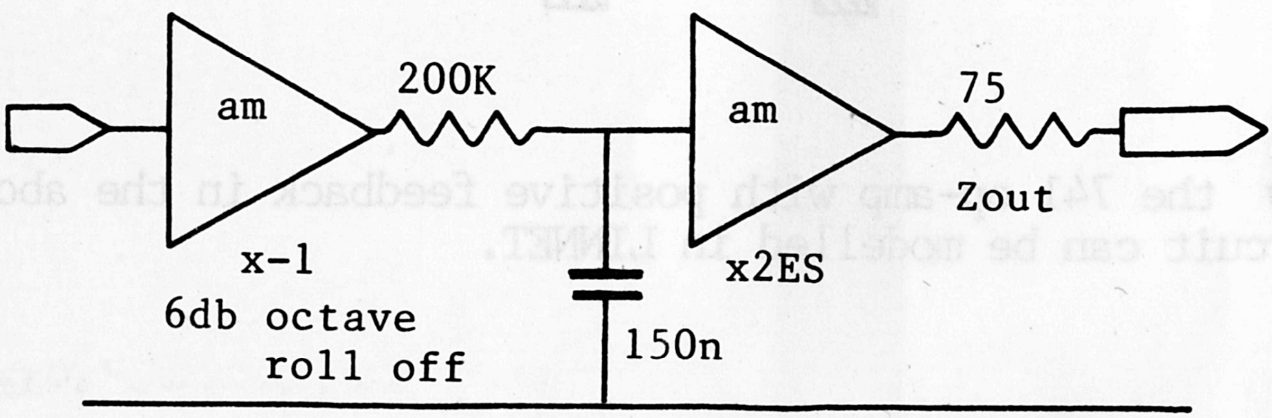


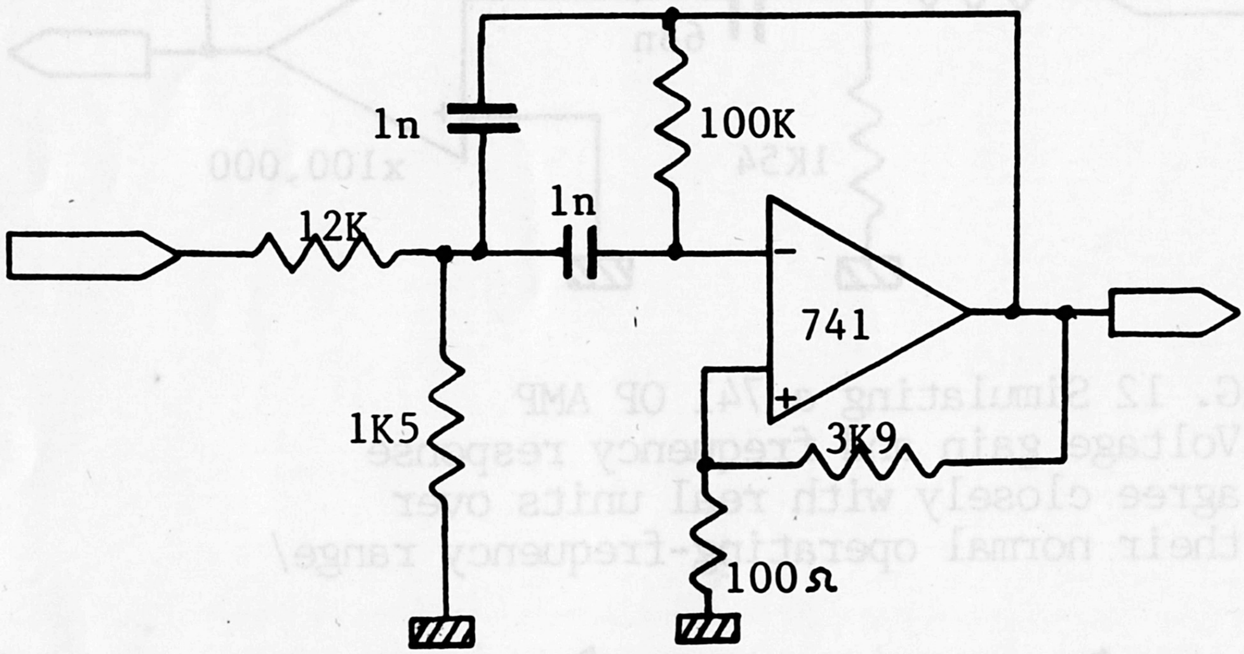
FIG. 12 Simulating a 741 OP AMP
Voltage gain and frequency response agree closely with real units over their normal operating-frequency range/



Comments:-
This element list assumes a near perfect op-amp. A more realistic result for a practical op-amp may be realised by using the model of fig.12. The element list then becomes:-

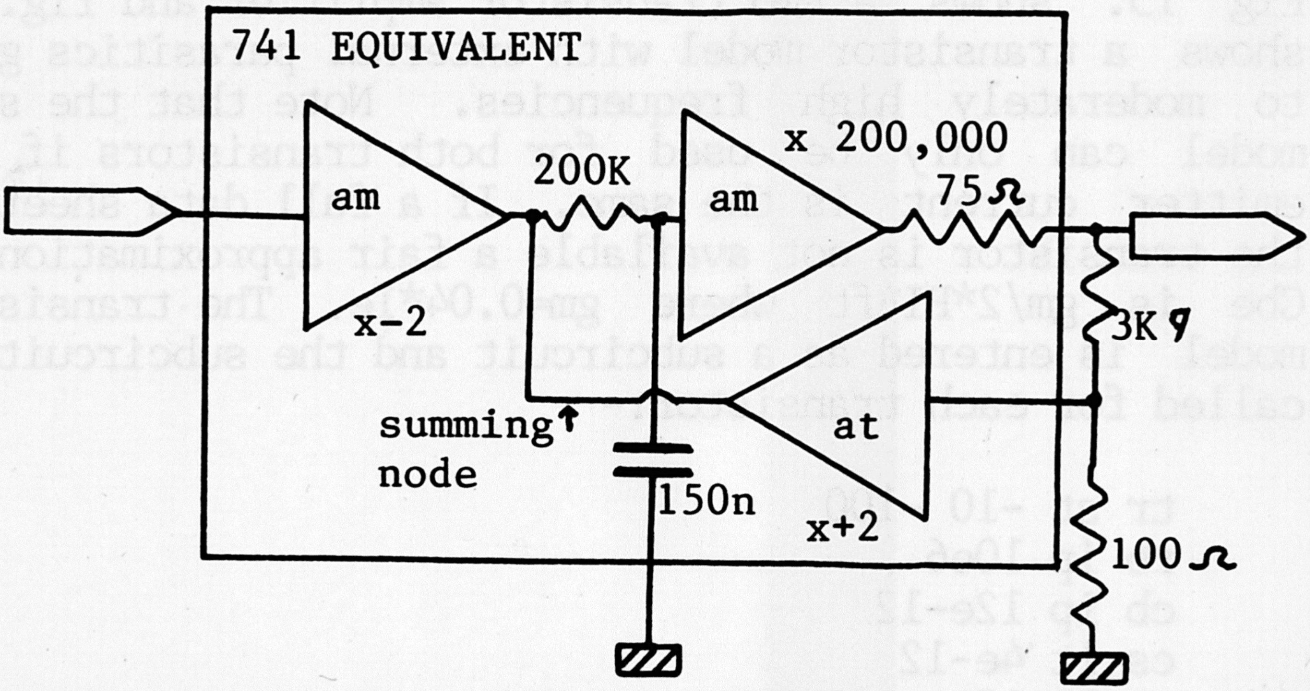
```
rb st 75
am lt 2e5
cs lt 150e-9
rb lt 2e5
am lt -1
rb ip 1e5
cb lt 68e-9
cb ip 68e-9
rs lt 1540
rb lt 12400
```

FIG. 13. OP AMP CIRCUIT WITH POSITIVE FEEDBACK.



How the 741 op-amp with positive feedback in the above circuit can be modelled in LINNET.

FIG.14.



OP AMP CIRCUIT WITH POSITIVE FEEDBACK

Fig.13 shows an op-amp circuit with positive and negative feedback. At first sight this seems difficult to encode for LINNET, however using the 741 model shown in Fig 14. and a subcircuit call enables us to encode it as follows:-

```
rb st 3900
rs lt 100
at lt 2
en pos_feedback
rb st 75
am lt 2e5
cs lt 150e-9
rb lt 2e5
cl pos_feedback ip
am lt -2
rb ip 1e5
cb lt 1e-9
cb ip 1e-9
rs lt 1500
rb lt 12e3
```

TWO TRANSISTOR AMPLIFIER

Fig 15. shows a two transistor amplifier and fig. 16 shows a transistor model with external parasitics good to moderately high frequencies. Note that the same model can only be used for both transistors if the emitter current is the same. If a full data sheet of the transistor is not available a fair approximation of Cbe is gm/2*PI*ft where gm=0.04*Ie. The transistor model is entered as a subcircuit and the subcircuit is called for each transistor:-

```
tr st -10 -100
rb ip 10e6
cb ip 12e-12
cs lt 4e-12
rb lt 15
en transistor
cl transistor st
cb ip 12e-12
rb ip 270e3
```

FIG. 15.

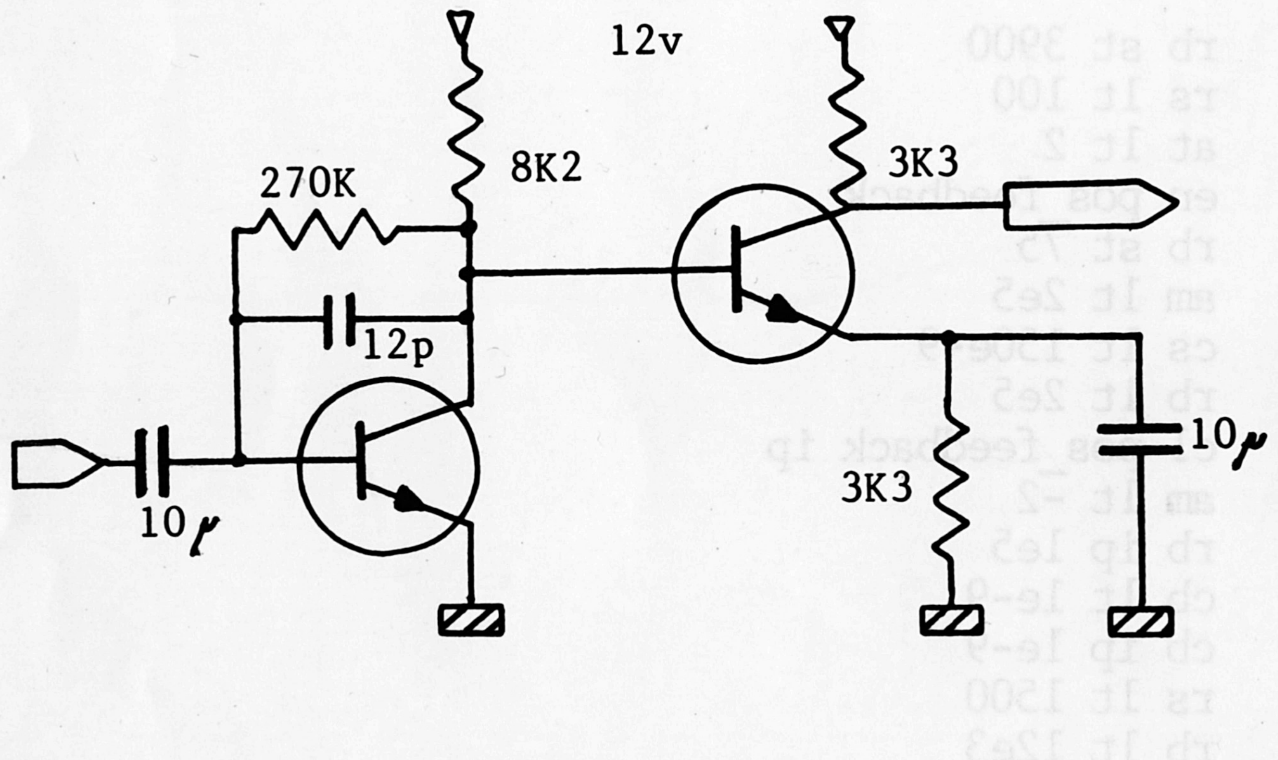
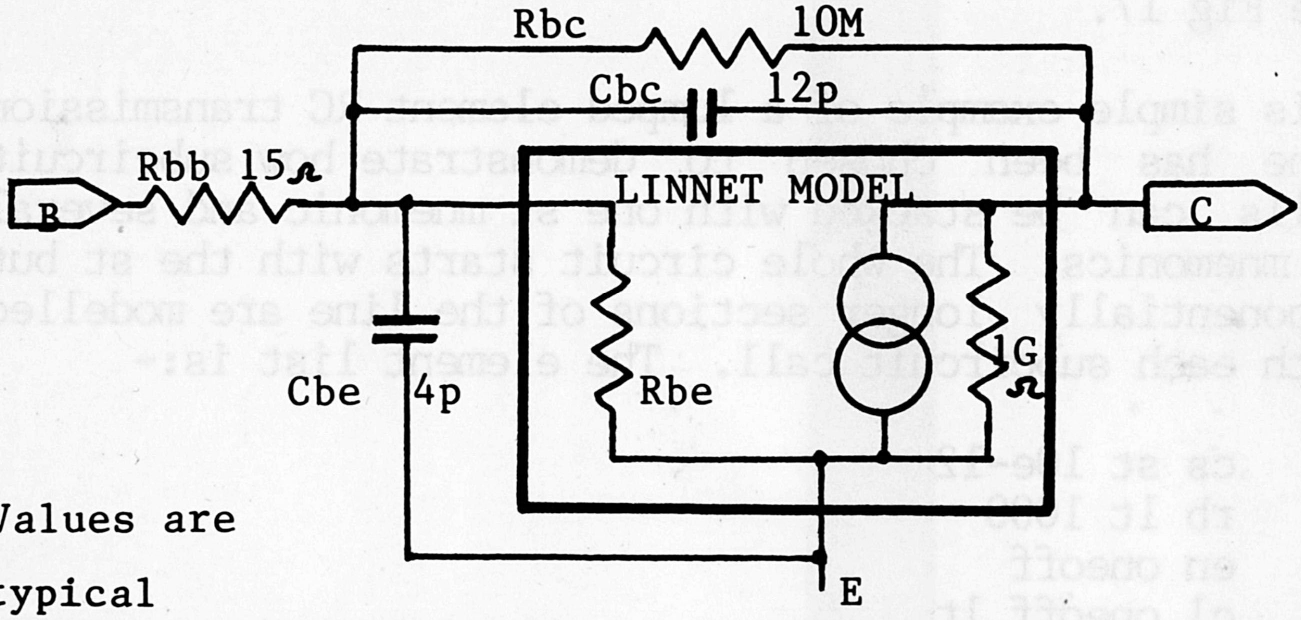


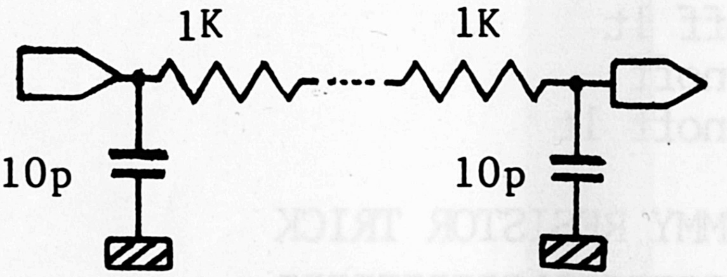
FIG. 16. HF TRANSISTOR MODEL



Values are

typical

FIG. 17 RC TRANSMISSION LINE.



****
Thirty-two identical sections

```
cb lt 10e-6
rs rt 8.2e3
en instage
rs st 3.3e3
cs lt 10e-6
cl transistor ab
rs rt 3.3e3
cl instage lt
```

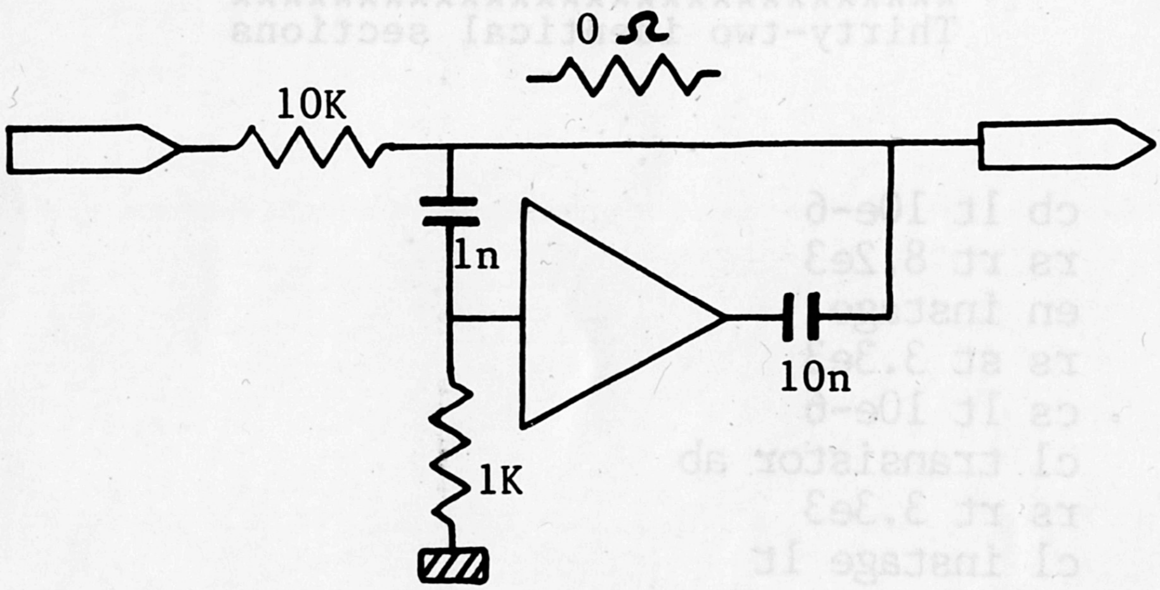
RC TRANSMISSION LINE

See Fig 17.

This simple example of a lumped element RC transmission line has been chosen to demonstrate how subcircuit calls can be stacked with one st mnemonic and several en mnemonics. The whole circuit starts with the st but exponentially longer sections of the line are modelled with each subcircuit call. The element list is:-

```
cs st 10e-12
rb lt 1000
en oneoff
cl oneoff lt
en twooff
cl twooff lt
en fouroff
cl fouroff lt
en eightoff
cl eightoff lt
en sixteenoff
cl sixteenoff lt
```

FIG. 18 THE DUMMY RESISTOR TRICK



THE ZERO RESISTANCE RESISTOR.

Fig. 18 shows a circuit that at first sight cannot be encoded for LINNET. However by considering one of the connections as a zero resistance resistor the encoding becomes straightforward as follows:-

```
cb st 10e-9
am lt 1
rs lt 1000
cb lt 100e-9
rb ip 0 ** dummy resistor **
rb lt 1e4
```

FIG. 19 CASCODE AMPLIFIER

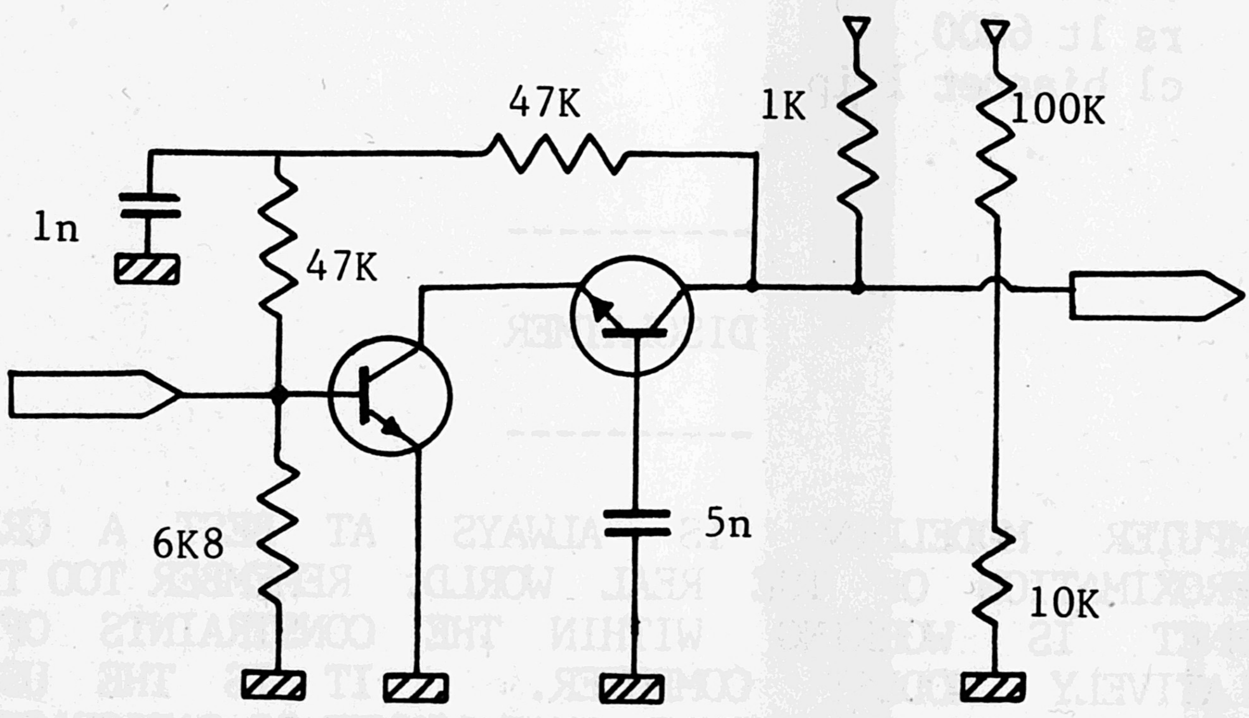


Fig.19 shows a two transistor amplifier using the cascode configuration. In the coding which follows it is left as an exercise for the reader to replace the transistor elements in the element list with calls to more suitable hf transistor models.

```
rb st 47000
cs lt 1e-9
rb lt 47000
en biasnet 1
rs st 100000
rs lt 10000
cs lt 5e-9
en biasnet 2
tr st 10 0.983
cl biasnet 2 be
rs rt 1000
tr lt -10 -130
rs lt 6800
cl biasnet 1 ip
```

DISCLAIMER

COMPUTER MODELLING IS ALWAYS AT BEST A CRUDE APPROXIMATION OF THE REAL WORLD. REMEMBER TOO THAT LINNET IS WORKING WITHIN THE CONSTRAINTS OF A RELATIVELY MODEST COMPUTER. IT IS THE USERS RESPONSIBILITY TO ENSURE THAT LINNET IS SATISFACTORY FOR THE PURPOSE FOR WHICH IT IS TO BE USED. IN PARTICULAR IN THE DESIGN OF CIRCUITRY WHERE PROPERTY, HEALTH OR LIFE MAY BE AT RISK LINNETS RESULTS SHOULD BE VERIFIED BY OTHER METHODS. THE AUTHOR DOES NOT ACCEPT ANY LIABILITY WHATSOEVER FOR THE CORRECTNESS OR OTHERWISE OF LINNETS OUTPUT OR FOR ANY COSTS INCURRED FROM ACTING ON SUCH OUTPUT.